Assignment_6

Shritej Chavan (BE14B004)

November 29, 2018

MLE and Least Squares

a) MLE method

The given two observations of the series are $x[1]$ and $x[2]$. An AR(1) model as follows

$$x[k] = -d_1 x[k-1] + e[k]$$

The parameters to be estimated are σ_e^2 and d_1 Assumption: $x[k]$ zero-mean stationary process Therefore,

$$E(x[k]) = E(x[1]) = E(x[2]) = 0$$

We know that the joint density function,

$$f(x, y) = f(x)f(y|x)$$

If all the density functions are conditional on another variable z just as in our case we can write it as follows

$$f(x, y|z) = f(x|z)f(y|x, z)$$

$$f(x[1], x[2]|\theta) = f(x[1]|\theta)f(x[2]|x[1], \theta)$$

Now calculating the variances and the expectation required for the above equation, we get

From ARMA model equation, we get

$$\sigma_x^2 = d_1^2\sigma_x^2 + \sigma_e^2$$

$$\sigma_{x[1]}^2 = \frac{\sigma_e^2}{1 - d_1^2}$$

$$E(x[2]|x[1]) = -d_1 x[1]$$

$$\sigma_{x[2]|x[1]}^2 = E((x[2]|x[1] - E(x[2]|x[1]))^2) = E(e^2[k]) = \sigma_e^2$$

To be obvious we know that joint distribution is gaussian, Therefore,

$$f(x[2]|x[1], \theta)$$

$$= (\frac{\sqrt{1 - d_1^2}}{\sqrt{2\pi\sigma_e^2}} exp(-0.5\frac{x^2[1](1 - d_1^2)}{\sigma_e^2})) * (\frac{1}{\sqrt{2\pi\sigma_e^2}} exp(-0.5\frac{(x[2] + d_1 x[1])^2}{\sigma_e^2}))$$

$$f(x[2]|x[1],\theta)=\frac{1}{2\pi\sigma_e^2}exp(-0.5\frac{x^2[2]+2d_1x[1]x[2]+x^2[1]}{\sigma_e^2})$$

Taking the log-likelihood, we get

$$L(\theta,x[2]x[1])=-ln(2\pi)+\frac{1}{2}ln(1-d_1^2)-2ln(\sigma)-\frac{(x^2[2]+2d_1x[1]x[2]+x^2[1])}{2\sigma_e^2}$$

Since we have two parameters $\theta=[\sigma_e^2,d_1]$ to estimate, therefore calculating the maximum likelihood by below solving equations,

$\frac{\partial L}{\partial\sigma_e^2}=0$ and $ = 0 $

We get,

$$\hat{d_1}=\frac{-2x[1]x[2]}{x^2[1]+x^2[2]}$$

$$\hat{\sigma_e^2}=\frac{(x^2[1]-x^2[2])^2}{2(x^2[1]+x^2[2])}$$

b) Least Squares

$$x[2]=-d_1x[1]+e[2]$$

The estimate of d_1 through least square solution is

$$\hat{d_1}=-\frac{x[2]}{x[1]}$$

Hence, we get

$$E((x[2]-\hat{x}[2])^2)=0$$

Hence, $\hat{\sigma}_e^2=0$.

For the given process comparing with MLE, we can say that Least squares is the better estimate.

Hannan-Rissanen Algorithm

For the Hannan-Rissanen algorithm, which is similar to the Durbin estimator where parameters are estimated linear least-squares regression of $v[k]$ on the estimated past innovations.

Where the past innovations are obtained as a residual after fitting a high order AR model

```
hr_fn = function(vk, ma, ar, max_ar){

 #Length of the series
```

```r
 N = length(vk)

 #Fitting a high order AR model
 arnmod = arima(vk, order = c(max_ar,0,0))

 #Extracting the residuals
 ek = arnmod$residuals

 #Omitting NA values
 ek = ek[which(is.na(ek)==FALSE)]

 ##Initializing z matrix
 z = matrix(NA,nrow=(N-(max(ma,ar))-1),ncol=(ar+ma))

 for (i in 1:ar){
   z[,i] = vk[(max(ar,ma)-i+2):(N - i)]

 }

 for(j in 1:ma){
   z[,j+ar] = ek[(max(ar,ma)-j+2) : (N-j)]

 }

#paramter set calculated using projection theorem

 theta_vec=((qr.solve(t(z) %*% z)) %*% t(z)) %*% vk[(max(ar,ma)+2):N]

 return(theta_vec)

}

#b)

#Simulating the MA series

ma2 = arima.sim(n = 100000, list( ma = c(1, 0.21)),sd = 1)

#Simulating the ARMA series

arma12 = arima.sim(n=100000, list( ar=c(0.4),ma = c(0.7, 0.12)), sd = 1)


mat_ma2 = matrix(data = NA, nrow = 10, ncol = 2)



for (i in 1:10){
```

```
  mat_ma2[i,] = t(hr_fn(ma2,2,0,i))
}

mat_ma2 = cbind(1:10, mat_ma2)
rownames(mat_ma2) = 1:10
colnames(mat_ma2) = c("Initial AR order", "c_1", "c_2")

mat_ma2

##    Initial AR order       c_1         c_2
## 1                 1 0.8248252 -0.02964988
## 2                 2 0.9050023  0.09643681
## 3                 3 0.9508055  0.15421753
## 4                 4 0.9747864  0.18203492
## 5                 5 0.9863281  0.19562354
## 6                 6 0.9918193  0.20189814
## 7                 7 0.9946074  0.20470759
## 8                 8 0.9960276  0.20635959
## 9                 9 0.9966657  0.20707908
## 10               10 0.9969806  0.20730653

#hr_fn(ma2,2,0,5)
mat_arma12 = matrix(data = NA, nrow = 10, ncol = 3)

#hr_fn(arma12,2,1,5)

for (j in 1:10){

  mat_arma12[j,] = t(hr_fn(arma12,2,1,j))
}

mat_arma12 = cbind(1:10, mat_arma12)
rownames(mat_arma12) = 1:10
colnames(mat_arma12) = c("Initial AR order","d_1", "c_1", "c_2")

mat_arma12

##    Initial AR order       d_1       c_1         c_2
## 1                 1 0.6456768 0.4347106 -0.26496786
## 2                 2 0.5183181 0.5656281 -0.04792544
## 3                 3 0.4439056 0.6522774  0.06656809
## 4                 4 0.4186996 0.6819118  0.10404178
## 5                 5 0.4133739 0.6881393  0.11218118
## 6                 6 0.4123385 0.6893108  0.11369907
## 7                 7 0.4120878 0.6895794  0.11407004
## 8                 8 0.4120985 0.6895589  0.11406706
## 9                 9 0.4122022 0.6894602  0.11394749
## 10               10 0.4121989 0.6894638  0.11395264
```

We can see that the estimates of the parameters get better with increasing order of the initial AR model, but reach a certain saturation point.

Also, with number data points required for simulating the process the parameters we are estimating gets better.

c) Now comparing the same with arma routine of the tseries package

```
library(tseries)
arma_pro = arima.sim(n = 100000, list( ar=c(0.4),ma = c(0.7, 0.12)),sd = 1)

ma_pro = arima.sim(n = 100000, list( ma = c(1, 0.21)),sd = 1)

armamod = arma(arma_pro, c(1,2))
armamod$coef

##         ar1         ma1         ma2   intercept
## 0.400518869 0.707611984 0.127165650 0.004394084

mamod = arma(ma_pro, c(0,2))
mamod$coef

##          ma1          ma2    intercept
##  0.995807157  0.205704320 -0.004444554
```

Spectral Densities

a) Theoretical spectral density

The given series formed by

$$x[k] = e[k-2] + 2e[k-1] + 4e[k]$$

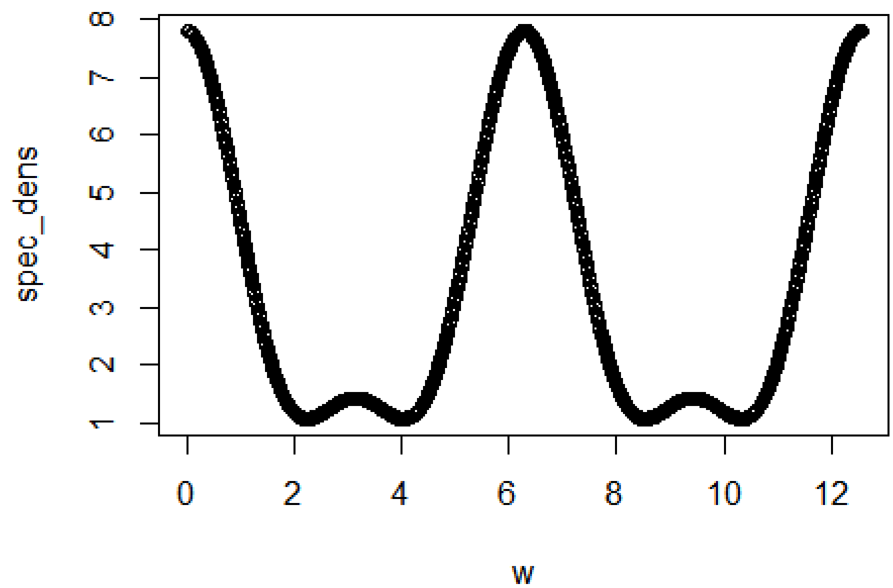
$$x[k] = (4 + 2q^{-1} + q^{-2})e[k] = H(q^{-1})e[k]$$

$$\gamma_{vv}(\omega) = |H(e^{-j\omega})|^2 \frac{\sigma_e^2}{2\pi}$$

Therefore,

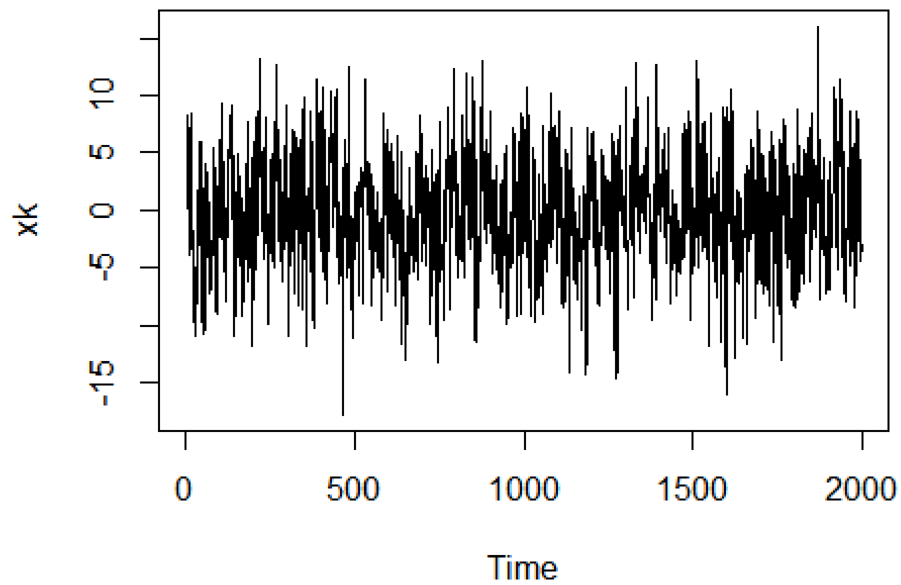
$$\gamma(\omega) = [8cos(2\omega) + 20cos(\omega) + 21]\frac{\sigma_e^2}{2\pi}$$

```
spec_dens = {}
w = seq(0,4*pi,0.01 )
for (i in 1:length(w)){
  spec_dens[i]=(21+(20*cos(w[i]))+(8*cos(2*w[i])))/(2*pi)
}
plot(w,spec_dens)
lines(w,spec_dens)
```

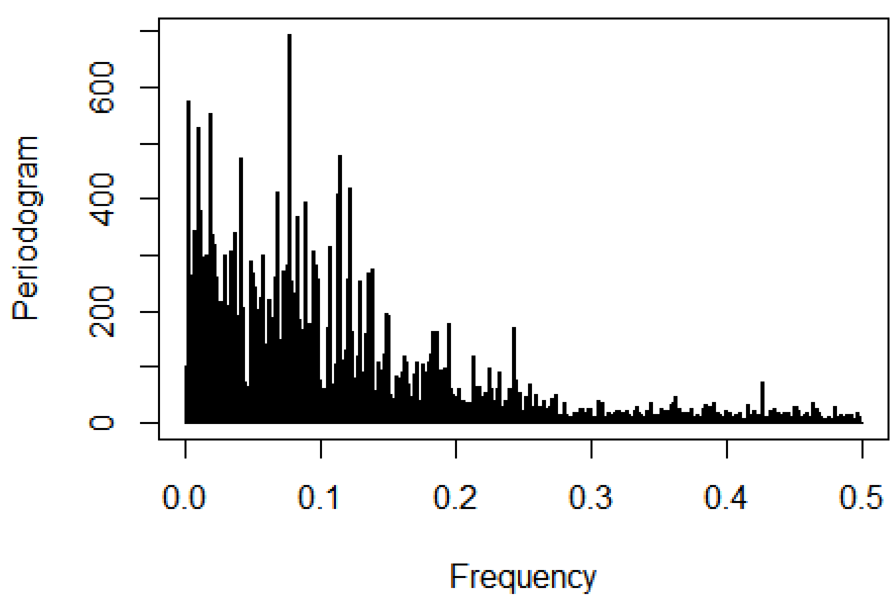


b) Generating x[k]

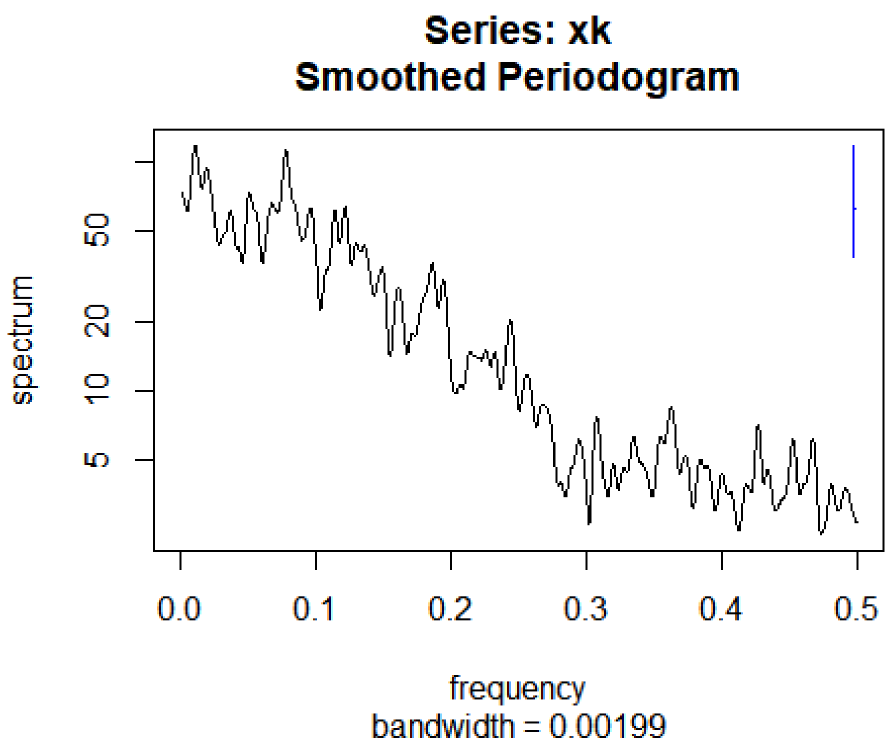
```
library(TSA)

##
## Attaching package: 'TSA'

## The following objects are masked from 'package:stats':
##
##     acf, arima

## The following object is masked from 'package:utils':
##
##     tar

xk = arima.sim(model = list(ma = c(4,2,1)), 2000)
plot(xk, type = 'l')
```



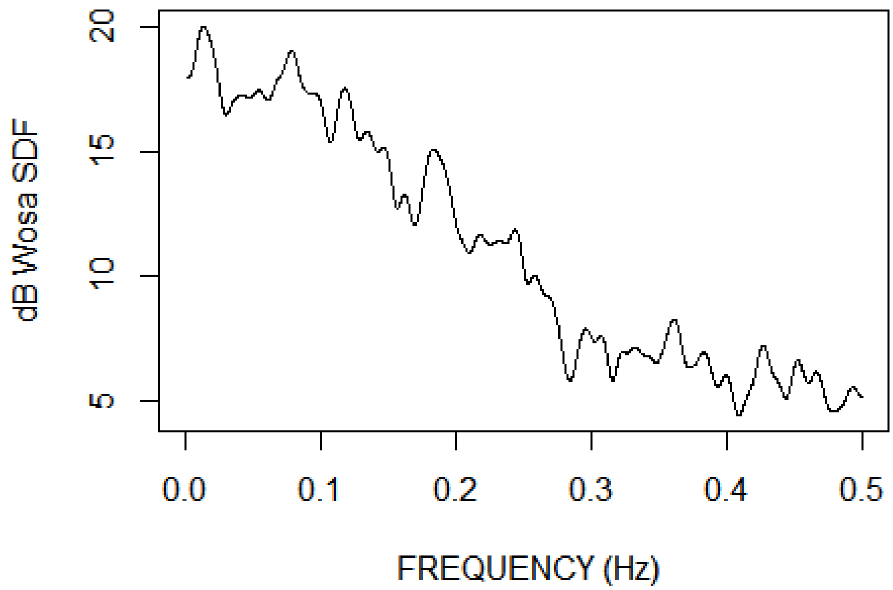

```r
# Raw Periodogram
periodogram(xk)
```



```r
#Daniell's Smoother
spec.pgram(xk, spans = rep(7,5))
```




```r
#Welch averaged periodogram

library(sapa)
welch = SDF(xk ,method="wosa" , blocksize= 150)
plot(welch)
```



iv) Parametric method

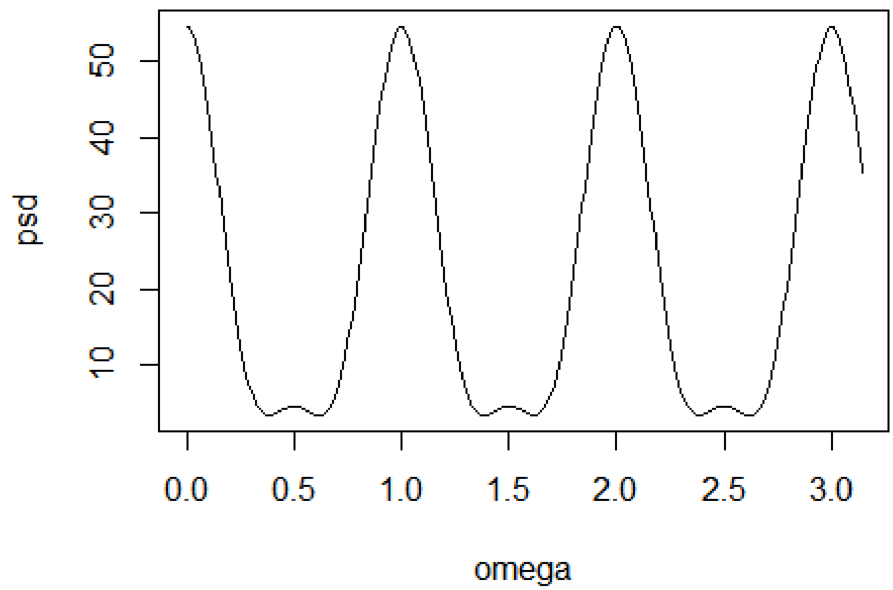
Estimating the parameters c_1 and c_2 by fitting the MA(2) model.

```
ma2mod = arima(xk, order = c(0,0,2))
c1 = ma2mod$coef[1]
c2 = ma2mod$coef[2]
```

Autocovariance functions for MA(2) models are as follows : $\sigma[0] = 1 + c_1^2 + c_2^2$ $\sigma[1] = c_1(1 + c_2)$ $\sigma[2] = c_2$ $\sigma[l] = 0 \forall l > 2$

Now doing fourier transform on the autocovariance functions to obtain Spectral Density we get

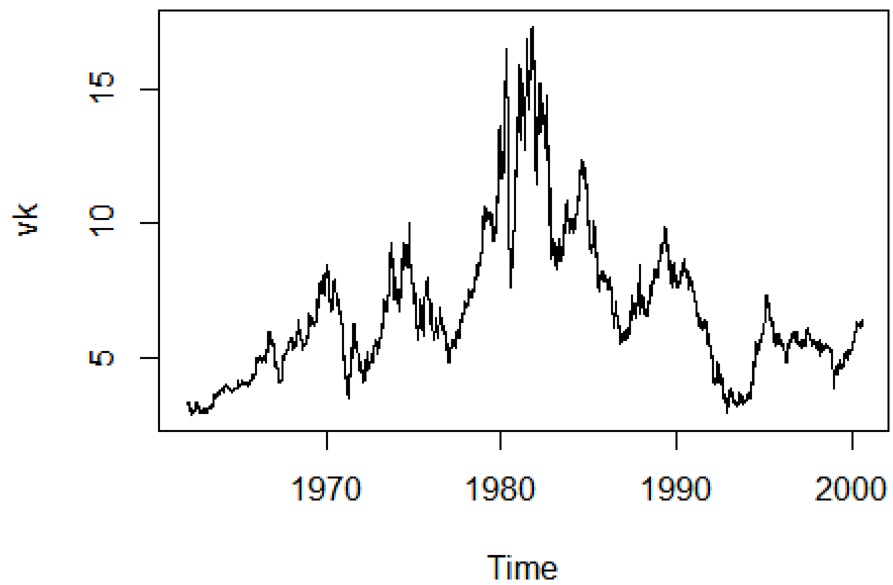
```
sig_e = ma2mod$sigma2

omega = seq(0,pi,by=0.01)

psd=
(sig_e)*((1+(c1^2)+(c2^2))+2*(c1+(c1*c2))*cos(2*pi*omega)+2*c2*cos(2*2*pi*ome
ga))


plot(omega,psd,type='l')
```



We can clearly see that the parametric method gives a better estimate of the Power spectral density than the other methods

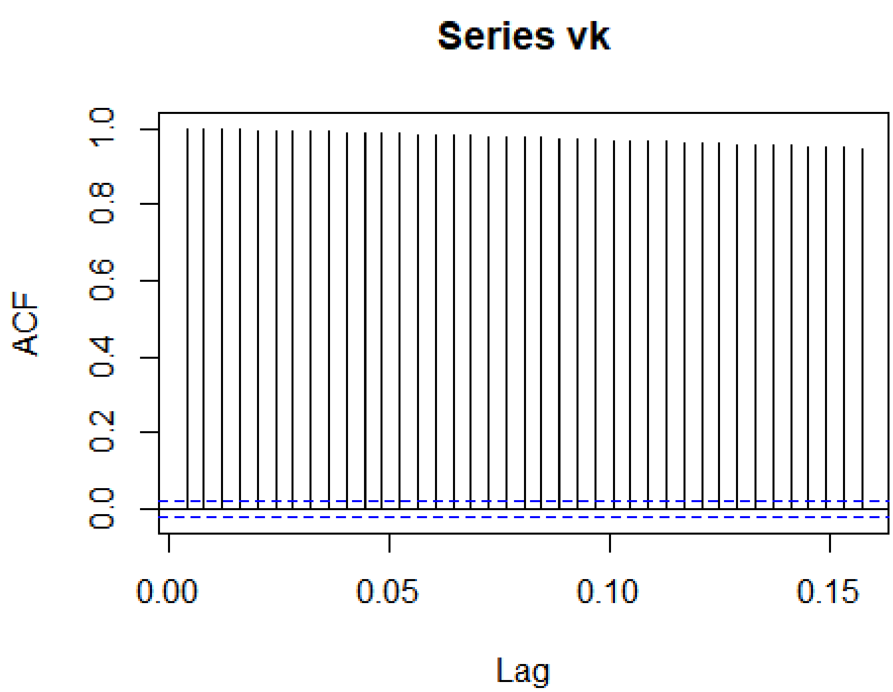
Time Series model

```
library(tseries)
library(TSA)
data(tcmd)

vk=tcm1yd
# The given data set

plot(vk)
```

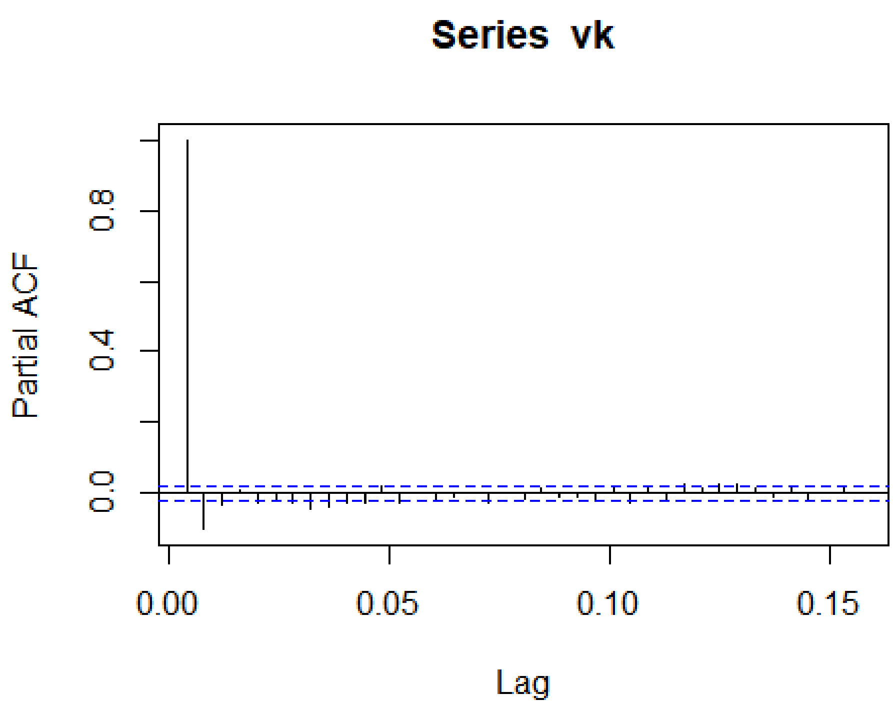


Now studying the ACF, PACF, we get

```
acf(vk)
```



```r
pacf(vk)
```



We can see that the acf of the series decays slowly which one of the characteristics of the integrating effects in the given series. Therefore differencing the series once we get

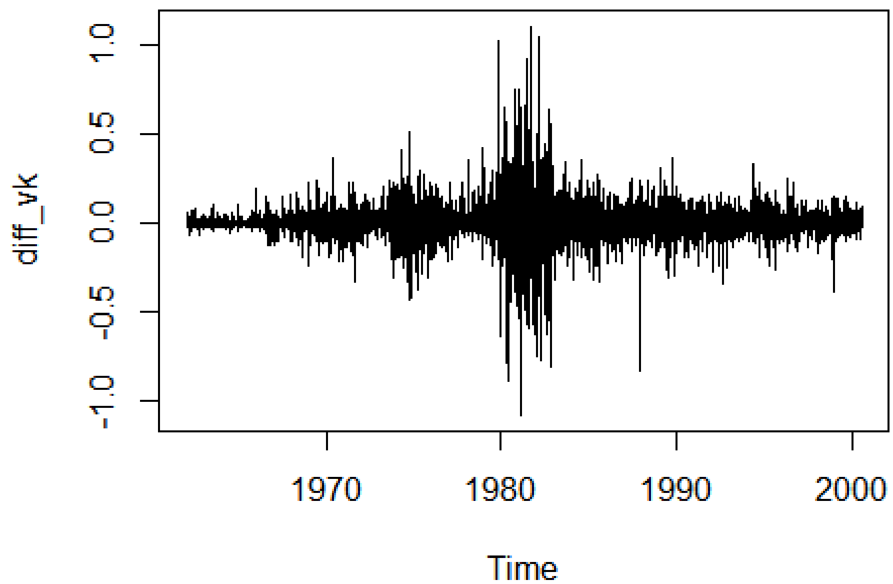
```r
diff_vk = diff(vk)
```

Now using Augmented Dick Fuller test to check the need for differencing, where null hypothesis states that differencing is required to obtain a stationary series.

```r
adf.test(diff_vk)

## Warning in adf.test(diff_vk): p-value smaller than printed p-value

##
##  Augmented Dickey-Fuller Test
##
## data:  diff_vk
## Dickey-Fuller = -17.642, Lag order = 21, p-value = 0.01
## alternative hypothesis: stationary
```

Therefore null hypothesis is rejected

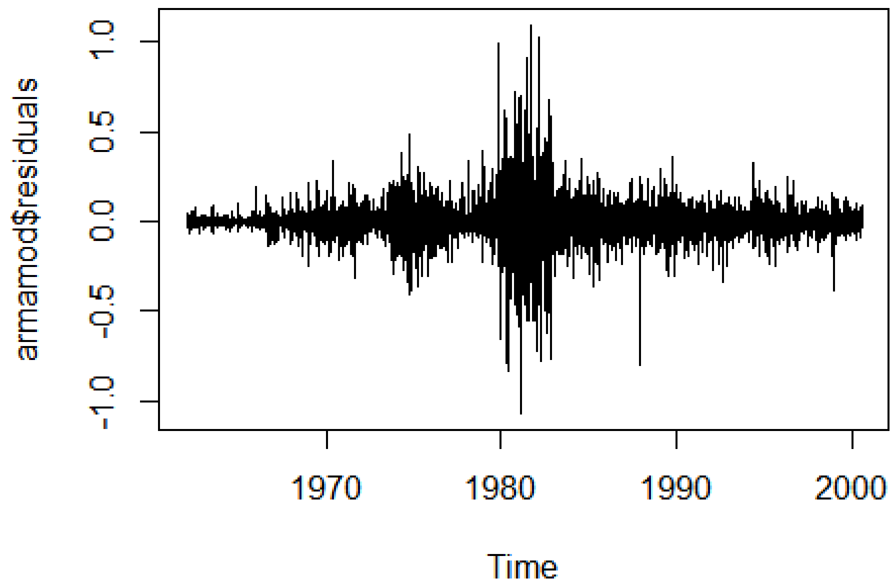
```r
plot(diff_vk)
```



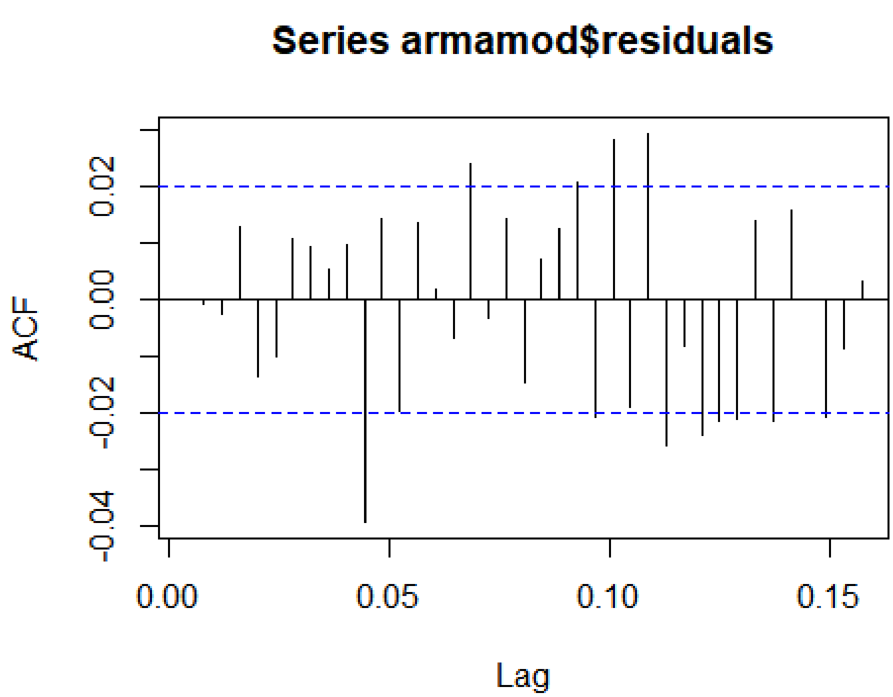
Now fitting the ARMA using the auto.arima function in the forecast package which searches through all the possible arma models and chooses the one with minimum aic value

```
library(forecast)

armamod <- auto.arima(diff_vk,seasonal =
FALSE,d=0,D=0,max.p=5,max.q=5,start.p = 1,start.q = 1)

summary(armamod)

## Series: diff_vk
## ARIMA(4,0,2) with zero mean
##
## Coefficients:
##          ar1      ar2      ar3     ar4      ma1     ma2
##       1.4950  -0.5920  -0.0062  0.0494  -1.3894  0.4740
## s.e.  0.1398   0.1414   0.0228  0.0115   0.1397  0.1255
##
## sigma^2 estimated as 0.009033:  log likelihood=8948.71
## AIC=-17883.43   AICc=-17883.41   BIC=-17833.26
##
## Training set error measures:
##                        ME       RMSE        MAE MPE MAPE      MASE
## Training set 0.0002196243 0.09501419 0.05393966 NaN  Inf 0.6605652
##                      ACF1
## Training set 0.0002009539
```

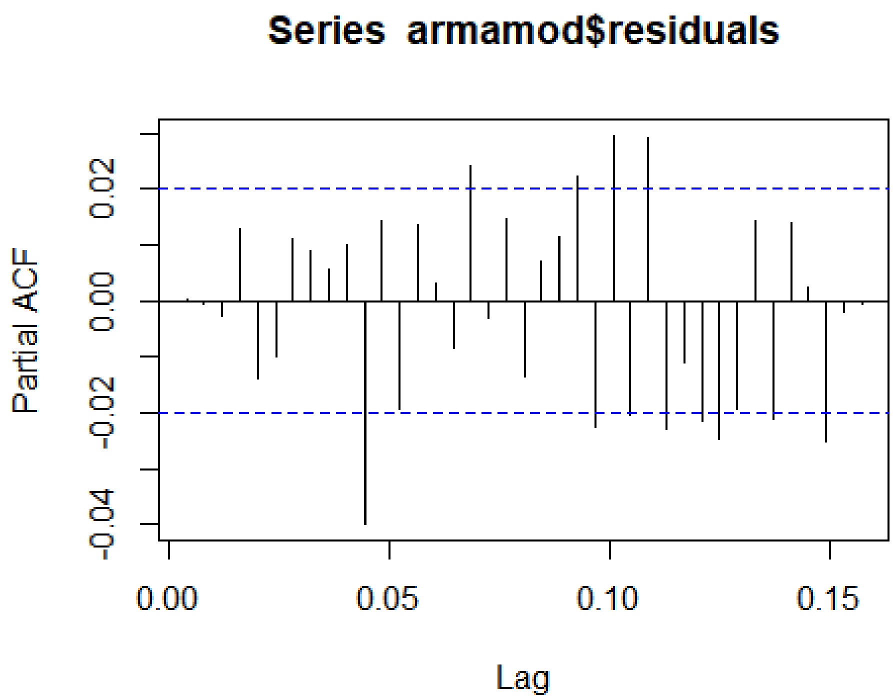
```r
plot(armamod$residuals)
```




```r
acf(armamod$residuals)
```



```
pacf(armamod$residuals)
```



ARMA model chosen by the function is ARMA(4,2) which is proper fit to the differenced series.

Predictions

For the given MA(1) process,

$$x[k] = e[k] - e[k-1]$$

For any process, the best prediction is the conditional expectation

We already proved that the conditional expectation of the RVs having joint distribution is the linear function of the independent RV.

Therefore in our case

$$\hat{x}[k+1|k] = \sum_{i=0}^{k-1} \hat{\Phi}[i]x[k-i]$$

Now,

Calculating the ACVF for the given MA (1) process, we get

$$\sigma[0] = E((x[k] - E(x[k])^2) = E(x[k]x[k])$$
$$= E(e[k]e[k]) - 2E(e[k]e[k-1]) + E(e[k-1]e[k-1])$$

$$\sigma[0] = \sigma_e^2 - 2(0) + \sigma_e^2$$

$$\sigma[0] = 2\sigma_e^2$$

Similarly we can calculate,

$$\sigma[1] = -\sigma_e^2$$

Also, $ = 0 l > 1 $

From Y-W's equations we get the following equations,

$$\hat{\Phi} = -\hat{\Sigma}_P^{-1} d_P$$

Where,

$$\hat{\Sigma}_P = \begin{pmatrix} 2 & -1 & 0 & \dots & 0 \\ -1 & 2 & -1 & \dots & 0 \\ 0 & -1 & 2 & \dots & 0 \\ \dots & \dots & \dots & \dots & 2 \end{pmatrix}$$

and

d_P the matrix of the coefficients of the process

$$d_P = (

)

$$ Therefore, solving for $\hat{\Phi}$ we get

$$\begin{pmatrix} \hat{\Phi}_1 \\ \hat{\Phi}_2 \\ \dots \\ \hat{\Phi}_k \end{pmatrix} = -\begin{pmatrix} \frac{k}{k+1} \\ \frac{k-1}{k+1} \\ \dots \\ \frac{1}{k+1} \end{pmatrix}$$

Therefore we get,

$$\hat{x}[k+1|k] = -\sum_{i=0}^{k-1} \frac{k-i}{k+1} x[k-i]$$

Calculating the mean square error in our above prediction of $x[k+1]$

By definition,

$$MSE = E((x[k+1] - \hat{x}[k+1|k])^2)$$

Sustituting the value of $\hat{x}[k+1|k]$, we get

$$MSE = E((x[k+1] + \sum_{i=0}^{k-1} \frac{k-i}{k+1} x[k-i])^2)$$

$$Mse = E(x^2[k+1] + 2x[k+1] \sum_{i=0}^{k-1} \frac{k-i}{k+1} x[k-i] + (\sum_{i=0}^{k-1} \frac{k-i}{k+1} x[k-i])^2)$$

Expectation of second term in the above equation will only be valid for lag =1

$$MSE = 2\sigma_e^2 + \frac{2k}{k+1}(-\sigma_e^2) + E(\sum_{i=0}^{k-1} \frac{(k-i)^2}{(k+1)^2} x^2[k-i] + \sum_{i=1}^{k-1} \frac{(k-i+1)(k-i)}{(k+1)^2} x[k-i]x[k-i+1])$$

$$MSE = \frac{2\sigma_e^2}{(k+1)} + \frac{2\sigma_e^2}{(k+1)^2}(\sum_{i=0}^{k-1}(k-i)^2 - \sum_{i=1}^{k-1}(k-i)(k-i+1))$$

Substituting $k - i = t$ in the first term and $k - i + 1 = t$ in the second term, we get

$$MSE = \frac{2\sigma_e^2}{(k+1)} + \frac{2\sigma_e^2}{(k+1)^2}(\sum_{t=1}^{k} t^2 - \sum_{t=2}^{k} t\,(t-1))$$

$$MSE = \frac{2\sigma_e^2}{(k+1)} + \frac{2\sigma_e^2}{(k+1)^2}(1 + \sum_{t=2}^{k} t)$$

$$MSE = \frac{2\sigma_e^2}{(k+1)} + \frac{2\sigma_e^2}{(k+1)^2} \sum_{t=1}^{k} t$$

$$MSE = \frac{2\sigma_e^2}{(k+1)} + \frac{2\sigma_e^2}{(k+1)^2} \frac{k(k+1)}{2}$$

$$MSE = \frac{(k+2)}{(k+1)} \sigma_e^2$$